HAL open science

Temporal Lifting Scheme for the Compression of Animated Sequences of Meshes

Frédéric Payan, Yasmine Boulfani-Cuisinaud, Marc Antonini

To cite this version:

Frédéric Payan, Yasmine Boulfani-Cuisinaud, Marc Antonini. Temporal Lifting Scheme for the Compression of Animated Sequences of Meshes. IEEE International Workshop VLBV 2005, Sep 2005, Italy. hal-00264648

HAL Id: hal-00264648
https://hal.science/hal-00264648

Submitted on 17 Mar 2008

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

TEMPORAL LIFTING SCHEME FOR THE COMPRESSION OF ANIMATED SEQUENCES OF MESHES

Frédéric Payan, Yasmine Boulfani, Marc Antonini
Laboratoire I3S - UMR 6070 CNRS / Université de Nice - Sophia Antipolis
Route des Lucioles - 06903 Sophia Antipolis FRANCE
Phone: +33 (0)4 92 94 27 22, Fax: +33 (0)4 92 94 28 98
{fpayan,am}@i3s.unice.fr

ABSTRACT

We present in this paper an original approach to encode the geometry of animated sequences of meshes (with a fixed connectivity). The main contribution is the use of a temporal lifting scheme to exploit the high temporal coherence in the geometry of successive frames. The sequences of wavelet coefficients are then encoded by an efficient coding scheme that includes a bit allocation optimizing the quantization process. Simulation results show that the proposed algorithm provides better compression performances than some state of the art coders.

I. INTRODUCTION

Today animated sequences of 3D meshes (figure 1) are more and more exploited to represent realistic visual data in many domains: computer games, character animation, physical simulations... This kind of data is often represented by a sequence of irregular meshes of fixed connectivity, and thus can be encoded independently by any compression method for static irregular meshes. Since the connectivity remains the same along the sequence, a more relevant approach is to consider the animated sequences of meshes as geometry deformations of one single static mesh (e.g. the first frame of the sequence). In order to exploit the temporal coherence, an efficient way to compress animated sequences is thus to encode the first frame, and then the position displacements of the vertices from frame to frame.

This can be done according to several techniques [1], [2], [3], [5], [6], [7], [8], [14]. For instance, we can cite the works of Alexa and Müller [2], who proposed a coding scheme based on the principal component analysis (*PCA*). Karni and Gotsman improved this method by further exploiting the spatial and temporal coherence and finally encode the *PCA* coefficients with a predictive coding scheme called *LPC* [7]. In parallel, Ibarria and Rossignac proposed a method which predicts the position of a vertex by using the already decoded vertices of its spatial neighborhood and its position in previous frames. Recently, Guskov and Khodakovsky presented a compression algorithm based on a multiresolution analysis [5]. In their work the analysis is applied on the geometry of each frame to obtain a multiresolution representation. A predictive coding scheme is then applied on the resulting details of each frame.

Fig. 1. Several frames of the animated sequence DOLPHIN.

In this paper, we propose an original approach based on a temporal lifting scheme to exploit the high temporal correlation between the geometry of successive frames. More precisely, we choose to exploit a monodimensional lifting scheme directly applied on the successive positions of the vertices across time.

The rest of this paper is organized as follows. Section II gives the principle of the proposed temporal lifting scheme for animated mesh sequences. Section III describes briefly the compression algorithm used. Simulation results and comparisons with some state of the art coders are shown in Section IV. We finally conclude and propose future works in V.

II. TEMPORAL LIFTING SCHEME FOR 3D MESH SEQUENCES

A. *Principle of the Lifting Scheme*

The lifting scheme is a second-generation wavelet transform that easily provides a multiresolution representation of signals, and enables decorrelation in space and frequency [12]. The idea of a 2-channel lifting scheme is to first split the original data in 2 subbands: the first subband contains the samples of odd indices and the second one contains the samples of even indices (figure 2). Two operators are then used: the *predictive* operator P is applied to obtain the subband of high frequency (*HF*) details (or wavelet coefficients), and then the *update* operator U is applied to obtain the low frequency (*LF*) signal (in other words a coarser representation of the original data).

There are a lot of lifting schemes, which depend on the dimension of the neighborhood used to compute the wavelet coefficients and the *LF* signal. So, a lifting scheme is generally defined by a pair $[n, m]$, where n and m are respectively the dimension of the prediction operator and the dimension of the update operator.

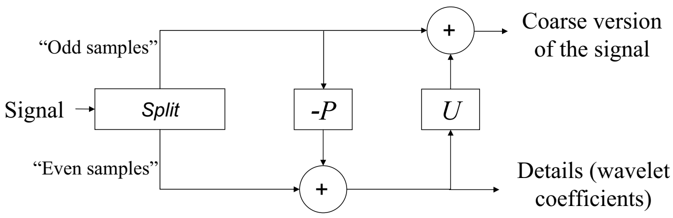

Fig. 2. 2-channel lifting scheme.

B. Temporal Lifting Scheme

The lifting scheme can be exploited to decorrelate data in space and frequency, but in case of signals with a temporal dimension, that is a signal $dD{+}t$ (d=1,2 or 3), a lifting scheme can be also applied along the time axis. This approach, called *temporal lifting scheme* allows to exploit the high temporal coherence existing in the processed data in order to reduce the information needed to represent the original sequence.

To our knowledge, the temporal lifting scheme has never been exploited in compression of such animated sequences of meshes (*i.e.* each frame of the sequence has the same connectivity). In the next section, we present how a temporal lifting scheme can be applied to exploit the temporal coherence of the geometry of an animated sequence of meshes.

C. Proposed Temporal Lifting Scheme

Formally, an animated mesh sequence is represented by a set of T static meshes $\{f_1, f_2, ..., f_T\}$, each frame having the same connectivity. As a lot of related papers [1], [2], [3], [6], [7], [8], [14], we consider the animated sequences of meshes as geometry deformations of one single static mesh (the first frame of the sequence). The main idea is thus to apply a monodimensional lifting scheme on the successive positions of each vertex of the mesh sequence (see figure 3). Hence, after a one-level decomposition, the sequence of T meshes is splitted in two subsequences of meshes:

- a sequence of $T/2$ *HF* detail sets denoted by h. These sets correspond to the frames of even index;
- the *LF* sequence of $T/2$ meshes denoted by l, that is a coarse version of the original one. These sets correspond to the frames of odd index.

By applying N times such a decomposition on the *LF* sequence previously computed, a multilevel decomposition is obtained, with N sequences of *HF* details $\{h^{(r)}\}$ (with r the resolution index), and a coarse version $l^{(N)}$ of the original sequence.

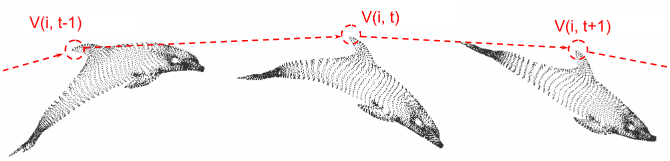

Fig. 3. Proposed approach. The previous and next positions of one vertex are used to compute the associated wavelet coefficient at the instant t. The matching of the treated vertex in different frames is implicit since the connectivity is similar for all the frames.

For instance, figure 4 shows the principle of the lifting scheme $[2, 0]$ (with 2 levels of decomposition) applied on the different positions of the i^{th} vertex. Once this process applied on each vertex, the whole *HF* sequences and the *LF* one are obtained.

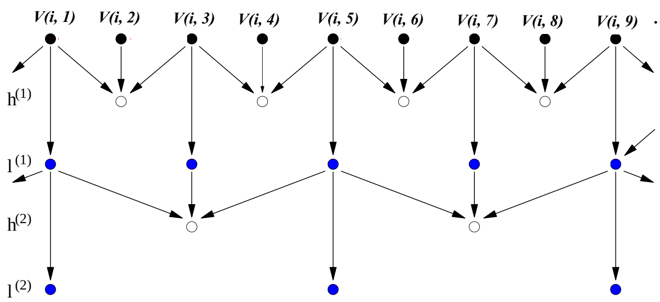

Fig. 4. A 2-level decomposition, with the lifting scheme $[2, 0]$. $V(i, t)$ is the position of the vertex i in the frame t.

III. Compression Algorithm

Once the temporal wavelet transform applied on the original sequence, the resulting sequences of wavelet coefficients and the *LF* one need to be encoded. For this purpose, we propose a compression algorithm similar to the coder proposed in [11], and presented in Figure 5.

The main feature of this coder is to include a bit allocation process optimizing the trade-off between the quality of the reconstructed sequence (after lossy compression and decompression) and the global bitrate. More precisely, this allocation process allows to compute the set of optimal quantizers, which minimizes the reconstructed mean square error for one specific user-given target bitrate.

As the wavelet transform is processed in parallel on the three coordinates of the vertex positions, we choose to encode separately the three sets of coordinates of each wavelet sequence with different non uniform scalar quantizers (SQ). Furthermore, it can be shown that the probability density function (PDF) of such a *HF* coordinate set can be modeled by a Generalized Gaussian Distribution (GGD), allowing a model-based allocation process [11].

Once the coefficients optimally quantized, the entropy coder presented in [10] is finally applied to provide the bitstream[1]. For more explanations, see [11].

Remark: Note that for the *LF* sequence the PDF of the three sets of coordinates cannot be modeled by a GGD. To overcome this problem, we use a *differential coding* similar to the technique presented in [9]. The idea is to encode the differences between the coordinates instead of the coordinates themselves. In [11], it has been shown that for static meshes the PDF of these differences can also be modeled by a GGD.

[1] In order to reconstruct the compressed data, the connectivity of the sequence must be also encoded and transmitted. Here, we use the connectivity coder of [13].

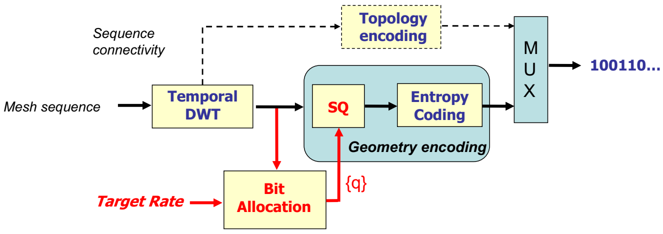

Fig. 5. Overview of the compression algorithm.

IV. Simulation Results

To evaluate the performances of the coder previously presented, we use the distortion measure introduced by Karni and Gotsman in [7], which corresponds to the relative discrete L_2-norm both in time and space. In the rest of the paper, this metric is called *KG error*, and is given in percent.

A. *Comparison of different Lifting Schemes*

First we compare the efficiency of the proposed coder in function of several lifting schemes. In this paper, we particularly deal with the schemes $[2, 0]$, $[2, 2]$, $[4, 2]$, and $[6, 2]$ [4]. Moreover, we always exploit a 4-level decomposition. For the comparison, we use two sequences with different features: FACE (10001 frames, 539 vertices), and CHICKEN (399 frames, 2916 vertices).

Figures 6 and 7 show the *KG error* in function of different user-given target bitrates for these two sequences. The bitrate is given in bits per vertex per frame. Globally, we observe that the proposed coder using the scheme $[2, 0]$ provides the worst coding performances, whereas the best coding performances are obtained by the schemes $[4, 2]$ or $[6, 2]$.

We point out that we obtain the expected results. Actually, the prediction operators of the schemes $[4, 2]$ and $[6, 2]$ "capture" more efficiently the vertex displacements than the schemes $[2, 0]$ and $[2, 2]$ since the latter take into account less neighbor samples. So, the prediction errors, *i.e.*, the wavelet coefficients are smaller, and the coding scheme is finally more efficient.

However, at low bitrates we observe for the sequence FACE that the schemes $[2, 2]$, $[4, 2]$ and $[6, 2]$ provide similar results. This is understandable, because globally this sequence does not present a high motion. The vertex displacements being small between successive frames, the prediction operators of the filters $[2, 2]$, $[4, 2]$ and $[6, 2]$ give almost the same wavelet coefficients. Furthermore, at low bitrates the coefficients are coarsely quantized. This involves that, for such a sequence with few motions the proposed coder gives similar coding performances whatever the filter used ($[2, 2]$, $[4, 2]$ and $[6, 2]$).

Finally, we showed on different sequences that the filters $[4, 2]$ and $[6, 2]$ provide similar results. So, we can conclude it is more relevant to use the filter $[4, 2]$ since the latter requires less computing resources in processing time and memory usage.

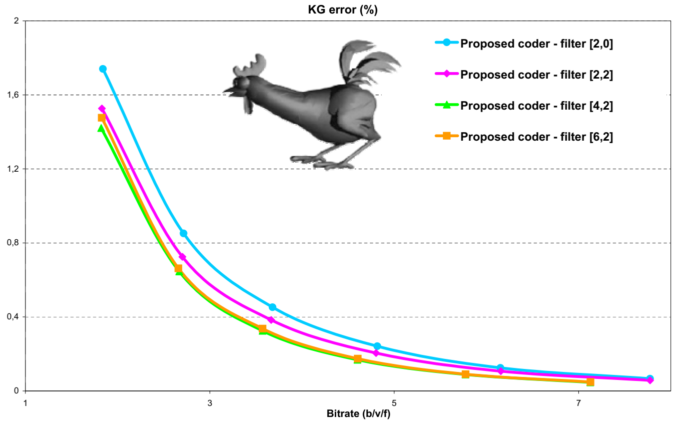

Fig. 6. Curve *KG error*/bitrate for the sequence CHICKEN according to different lifting schemes.

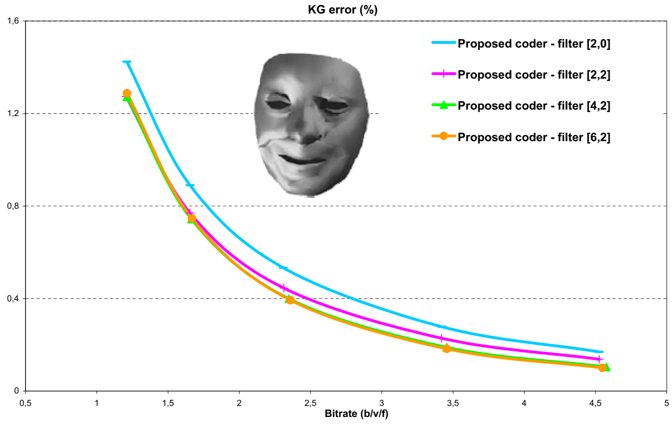

Fig. 7. Curve *KG error*/bitrate for the sequence FACE according to different lifting schemes.

B. *Comparison with some State of the Art Coders*

In this section we compare the coding performances of the proposed coder with some state of the art coders:

- the coder for static meshes of Touma and Gotsman [13] denoted by *TG*;
- the PCA-based coder for mesh sequences of Alexa and Müller [2] denoted by *PCA*;
- the coder for mesh sequences of Karni and Gotsman [7] denoted by *KG*. It includes the *PCA* approach of [2] and a linear prediction coding;
- the coder *Dynapack* of Ibarria and Rossignac [6].

The results relative to these methods are extracted from [7]. Therefore some results are missing.

Figure 8 shows that for the sequence FACE our method is more efficient than the methods *TG* and *Dynapack*. On the other hand, the methods *PCA* and *KG* provide significant better coding performances than our algorithm. This is understandable given the specific features of this sequence. The *PCA*-based methods are indeed particularly performant when the number of frames is much higher than the number of vertices [7].

In parallel, for the sequence CHICKEN (Figure 9) we observe that our method is significantly better than *TG* but also

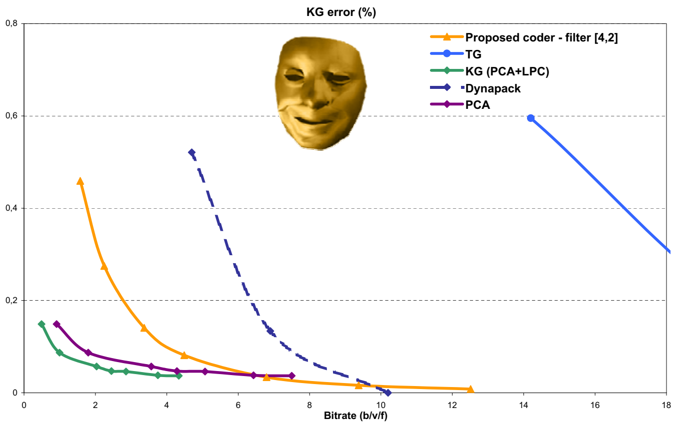

Fig. 8. Curves *KG Error*/bitrate for the sequence FACE relative to different compression methods.

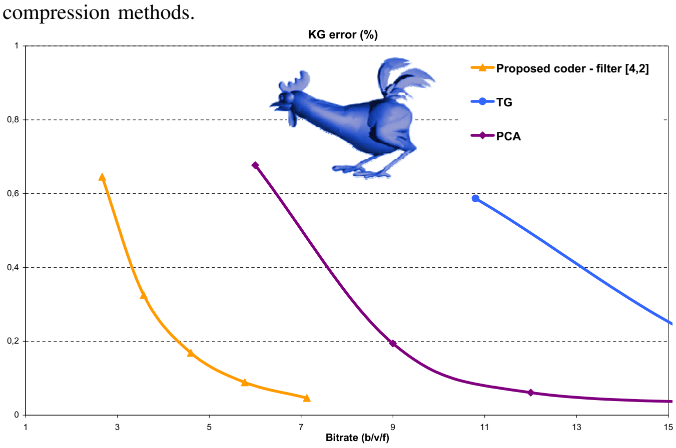

Fig. 9. Curves *KG Error*/bitrate for the sequence CHICKEN relative to different compression methods.

better than *PCA*. Contrary to FACE, the sequence CHICKEN has a much smaller number of frames than vertices. It is recognized that the *PCA*-based methods are not suitable for such cases [7], but this is not a drawback for the proposed method.

In addition, the simplicity of the lifting scheme process is an high advantage compared to the complex analysis process used by the coders *PCA* and *KG* to compute the sets of details [7].

V. CONCLUSIONS AND FUTURE WORKS

In this paper, we have introduced a temporal lifting scheme for the geometry of animated mesh sequences (with a fixed connectivity), the final objective being an efficient wavelet-based compression algorithm for such data. The advantage of this lifting scheme is to strongly reduce the information needed to represent the geometry of a mesh sequence by exploiting its temporal coherence. We have shown experimentally that the filters $[4, 2]$ and $[6, 2]$ are the most efficient ones.

Furthermore, we have shown that the proposed temporal lifting scheme associated to an optimal coder (that includes a model-based bit allocation optimizing the quantization process) provides better coding performances than several state of the art coders, in particular when the number of frames is much smaller than the number of vertices of each frame.

In addition, even if the proposed coder should not always provide better results than the coders based on a *PCA* approach, it has the advantage to be much faster, and its efficiency does not depend on the features of the sequence.

There are a lot of perspectives for such a coding scheme. The first one is about the *LF* data encoding, which is at present suboptimal. For instance, using an additional *temporal prediction coding* on the successive *LF* frames should significantly improves the coding performances.

ACKNOWLEDGEMENTS

The sequence CHICKEN is the property of Microsoft Inc. and the sequence FACE was kindly generated by Demetri Terzopoulos. We are particularly grateful to Zachi Karni for providing us with these data. All the results of the state of the art methods are extracted from the paper of Karni and Gotsman [7].

REFERENCES

[1] J.H. Ahn, C.S. Kim, C.C.J. Kuo, and Y.S. Ho. Motion compensated compression of 3D animation models. *IEEE Electronics Letters*, 37(24):1445–1446, Nov 2001.
[2] M. Alexa and W. Müller. Representing animations by principal components. *Comput. Graph. Forum 19,*, 3, 2000.
[3] H.M. Briceno, P.V. Sander, L. McMillan, S. Gotler, and H. Hoppe. Geometry videos: a new representation for 3D animations. *ACM Symp. Computer Animation*, pages 136–146, 2003.
[4] A.R. Calderbank, I. Daubechies, W. Sweldens, and B.L. Yeo. Wavelet transforms that map integers to integrers. *Applied and Computational Harmonic Analysis*, 5(3):332–369, 1998.
[5] I. Guskov and A. Khodakovsky. Wavelet compression of parametrically coherent mesh sequences. In *Eurographics/ACM SIGGRAPH Symposium on Computer Animation*, August 2004.
[6] L. Ibarria and J. Rossignac. Dynapack: space-time compression of the 3D animations of triangle meshes with fixed connectivity. *ACM Symp. Computer Animation*, pages 126–135, 2003.
[7] Z. Karni and C. Gotsman. Compression of soft-body animation sequences. *Computers and Graphics*, 28:25–34, 2004.
[8] J. E. Lengyel. Compression of time-dependent geometry. In *ACM Symposium on Interactive 3D Graphics*, pages 89–96, 1999.
[9] F. Payan and M. Antonini. Multiresolution 3D mesh compression. In *Proceedings of IEEE International Conference in Image Processing (ICIP)*, Rochester, USA, september 2002.
[10] F. Payan and M. Antonini. 3D multiresolution context-based coding for geometry compression. In *Proceedings of IEEE International Conference in Image Processing (ICIP)*, Barcelona, Spain, sept. 2003.
[11] F. Payan and M. Antonini. An efficient bit allocation for compressing normal meshes with an error-driven quantization. *Computer Aided Geometric Design, Special Issue on Geometric Mesh Processing*, 22:466–486, July 2005.
[12] Wim Sweldens. The lifting scheme: A construction of second generation wavelets. *SIAM Journal on Mathematical Analysis*, 29(2):511–546, 1998.
[13] C. Touma and C. Gotsman. Triangle mesh compression. *Graphics Interface'98*, pages 26–34, 1998.
[14] J.H. Yang, C.S. Kim, and S.-U. Lee. Compression of 3D triangle mesh sequences based on vertex-wise motion vector prediction. *IEEE Trans. Circuits Syst. Video Technol.*, 12(12):1178–1184, Dec. 2002.